User Manual for Cognitive Vision Software

Current CoViS Group
Emre Baseski, Sinan Kalkan, Lars Baunegaard With Jensen,
Anders Kjr-Nielsen, Dirk Kraft, Norbert Krüger,
Florian Pilz, and Nicolas Pugeault, Morten Skov

Further people that have been involved in CoViS:
Michael Felsberg, Oliver Granert, Thomas Jäger,
Martin Pörksen, Christian Gebken, Morten Kjaergaard,
Mila Popovic, Mogens Christensen,
Jesper Juul Henriksen, Hengfuan Lin,
Yan Shi, Shuang Zhang, Markus Ackermann,
Marco Hahn and Daniel Wendorff.

Contents

1 Introduction **3**
- 1.1 Overview . . . 3
- 1.2 Status . . . 3
- 1.3 History and Philosophy . . . 3

2 Requirements **5**
- 2.1 Operating systems: . . . 5
- 2.2 GCC and Auto tools . . . 5
- 2.3 Basic libraries . . . 5
- 2.4 FFTW and MFFT . . . 5
- 2.5 Numa . . . 6
- 2.6 Cairo (Optional) . . . 6

3 Installation guide **6**
- 3.1 How to retrieve the code from the CVS server . . . 6
- 3.2 Compiling MoInS . . . 7
- 3.3 Installing FFTW and MFFT libraries . . . 7

4 Directory Structure **8**
- 4.1 MoInS . . . 8
- 4.2 Source . . . 8
  - 4.2.1 Devices . . . 8
  - 4.2.2 Features . . . 8
  - 4.2.3 Image . . . 8
  - 4.2.4 Matching . . . 8
  - 4.2.5 Mathematics . . . 8
  - 4.2.6 MGraphics . . . 9
  - 4.2.7 Relations . . . 9
  - 4.2.8 Representation . . . 9
  - 4.2.9 tclap . . . 9
  - 4.2.10 Utilities . . . 9
- 4.3 Demo . . . 9

5 Doxygen Documentation **9**

6 Demo Applications **10**
- 6.1 Capturing Image Sequences Demo . . . 10
- 6.2 Accumulation Demo . . . 10
- 6.3 Rigid Body Motion Estimation Demo . . . 10
- 6.4 Wanderer Demo . . . 10

7 The ten Commandments **11**

8 Publications based on or related to CoViS **11**
- 8.1 Publications . . . 11

1 Introduction

1.1 Overview

This manual gives an introduction into the Software Library CoViS (**Co**gnitive **Vi**sion **S**oftware) which is maintained by the Cognitive Vision Lab [6]. The development of CoViS has started in 1998 as a framework in which bachelor and master theses become realised.

Having the aim to develop cognitive vision based system it was obvious that such an attempt can not be approached within a single master or PhD thesis but that it requires the interaction of sub–modules derived by a group scientists. Meanwhile (Status November 2006) within CoViS more than 20 Bachelor and Master theses have been realised (for a full list of publications, PhD and master theses see [6]). Furthermore, two PhD theses are close to be submitted and a number of PhD and master students are currently working within the CoViS framework. CoViS has been developped and used in the three European projects ECOVision [2], Drivsco [3] and PACO+ [1]. The scientific content has been published in 12 journal and more than 30 conference publications. These publications as well as the PhD, master and bachelor theses give a good understanding of the different sub–modules.

Within CoViS a particular visual representation has been created that. This representation is based on so called 'multi–modal primitives' that are the building block an early cognitive vision system described in, e.g., [7, 5]. This representation has received some attention from other vision groups and this document is supposed to make it easier for people outside CoViL or new members of CoViL to become familar with the basic modules and their applications.

In section 2 we describe requirements for the installation of CoViS while in section 3 we give some help for its installation. In section 6, four demo programs show the processing of basic sub-modules of the software on available data.

In section 7 we mention ten basic Coding conventions. When you not only use CoViS but also do programming in it you should read these.

1.2 Status

We are currently determining the legal status of CoViS. We will make it a public domain software and are currently looking in the issue of an appropriate software license.

1.3 History and Philosophy

One reason for the particular strength of the human visual and cognitive system comes from its ability to integrate and reason about information from different visual or sensorial modalities. Naturally this leads to the necessity of data transfer between sub–modules which requires special care in the design of software as well as the working environment in a group of researchers. Hence, working within one software library was a natural decision. Nevertheless, the

additional efforts that are required to maintain such a structure should not be ignored: First, it requires a certain discipline of programmers to maintain a certain quality of programming that allows for further use. Second, it requires the willingness to accept certain compromises that have been made during the development. Third, a person not having been involved in the programming framework needs to allow for some time for him/her 'to get into the code'. Fourth, a certain team spirit as well as the ability to discuss with and learn from others is required.

Not neglecting such potential problems and in the process of finding solutions for these, we nevertheless believe that there are good reasons to go on with our approach. First, we see no other option anyway since it is unlikely that truely cognitive systems can be written in a one person attempt. The underlying scientific problems are simply to hard and various. Secondly, the necessity to communicate within the group — although requiring a feeling for mutual responsibility as well as a general openess and commitment towards colleagues — increases the joy of working. Thirdly, it is good to know that the wheel has not to be invented all times again but that significant progress can be made by using existing sub–modules. In vision and robotoc science we are sitting on the shoulder of giants anyway.

CoViS has been initiated in 1998 by Norbert Krueger and his master students Markus Ackermann, Marco Hahn and Daniel Wendorff at the Cognitive System Group at the University of Kiel in Germany. The original name was KiViGraP for Kieler Vision and Graping Project and the user will still find some documentation in German in the code. Further contributers to CoViS in Kiel have been Michael Felsberg, Oliver Granert, Thomas Jäger, Martin Pörksen and Christian Gebken.

In 2001 Norbert moved to Stirling in Scotland to the group of Florentin Wörgötter where CoViS (under the name of MoInS abbreviating 'Modality Integation Software') became part of the European project ECOVision [2]. While in Kiel research was focussed on robotics and vision the development of CoViS was only concerned with vision. Nicolas Pugeault and Sinan Kalkan have been contributing in their PhD leading to the currently existing representations.

In 2003 Norbert accepted a faculty position at the University of Aalborg and initiated the Cognitive Vision Lab. Florian Pilz (PhD student), Lars B. Christensen, Brian Wettegren, and Thomas Kiesche (master) as well as Yan Shi (bachelor) have been contributing to CoViS. Beginning of 2006 the two European projects Drivsco and PACO+ started and more PhD/RA students became involved. Dirk Kraft, Emre Baseski and Morten Skov have been contributing to CoViS.

In 2006, Norbert changed to the Maersk Institute at Syddansk University which offers with its international robotic master a suitable educational platform for the further development of CoViS. The project that we are currently mainly addressing are concerned with vision in a robotic environment and by that lead back to the original starting point of KiViGraP.

Since the purpose of CoViS has not been the development of a library being used in a company but mainly to be a framework in which bachelor, master

and PhD theses become preserved, the availablity of documentation is rather varying depending on the individual programmer. This needs to be counted as an disadvantage compared to software standards in companies that can put a lot on manpower in documentation. However, the scientific content is well doscumented in the different theses and publications and we are currently working on the improvement of documentation and structure and this manuel and the demo programs are one step in this direction.

2 Requirements

2.1 Operating systems:

The software is known to compile well on linux systems (kernel 2.4 or 2.6) such as recent versions of SuSE, Debian, Ubuntu and Gentoo. It can compile under Mac OS 10.3 but requires detailed technical knowledge of the system. The standard distribution used by the group is Ubuntu and as such most support can be expected in the Ubuntu environment.

2.2 GCC and Auto tools

Currently the software compiles with gcc-4.0.0 (supporting also earlier gcc versions like 3.3), automake version 1.7 and newer. Autoconf works for any newer versions.

2.3 Basic libraries

- Libtiff-devel
- Libjpeg-devel
- libglib-devel
- X11-dev
- libs
- glut-dev
- libxml++-dev
- raw1394-dev
- cairo2-dev

2.4 FFTW and MFFT

These packages are provided by the software package. They are not required, but highly recommended to install to ensure fast FFT computations.

2.5 Numa

This package is required to ensure compatibility with future versions. The Numa API controls scheduling and memory placement policies on multi-core architecture systems.

2.6 Cairo (Optional)

Cairo can be used to obtain neater and faster output in images. For example this can be enabled in the Accumulation demo.

3 Installation guide

A step-by-step guide on installation and installation issues. This will include information on the various approaches of getting hold on the software. This guide can also be found at http://covig.imi.aau.dk/installation.html

3.1 How to retrieve the code from the CVS server

- Make sure you have a cvs client installed on you system
- Add following enviroment variables (e.g in .profile (SuSE) or .bashrc)

  ```
  export CVS_RSH="ssh"
  export CVSROOT=":ext:username@134.76.221.181:/home/moins/cvs"
  ```

  username is to be replaced by your own cvs username.

- Create a directory for MoInS in your home:

  ```
  mkdir /home/username/MoInS
  ```

- Do an inital checkout by typing

  ```
  cvs co MoInS
  ```

  For the latest stable version, you need to type:

  ```
  cvs checkout -r public_1_0 MoInS
  ```

  The cvs server asks for you cvs password and copies all files onto your machine.

3.2 Compiling MoInS

- Add following enviroment variables (in .bashrc)

  ```
  export KIVI_HOME="/home/username/MoInS"
  export LD_LIBRARY_PATH="/home/username/MoInS/lib"
  ```

  If your MoInS root directory or library path is different then change the paths accordingly.

- Check the configure.in file in the MoInS root directory and adjust compiler settings to your needs. A good start would be:

  ```
  CXXFLAGS="-ggdb -O2"
  ```

  But beware that `-O2` option makes it not possible to access file and line number information in gdb.

- Run the linux script in the MoInS root directory to compile all the source.

3.3 Installing FFTW and MFFT libraries

- Download fftw version 3.x (www.fftw.org)
- Installing the fftw (Read the README file included)

  ```
  ./configure --prefix=/usr --enable-shared
  make
  sudo make install
  ```

- Download mfft version 0.2.4 (http://www.covig.imi.aau.dk/mfft/)
- installing the mfft

  ```
  ./configure --prefix=/usr
  make
  sudo make install
  ```

- Create your own mfft configuration file. See sample.conf in mfft the source directory
- Add default location for mfft config file in .bashrc

  ```
  export MFFT_CONFIG="/home/username/path_to_my/mfft.conf
  ```

- Execute the following command

  ```
  source .bashrc
  ```

- Add the option –enable-mfft in the linux script and recompile MoInS

4 Directory Structure

4.1 MoInS

```
CVS
[CVS Information]
Demo
[Demo programs]
Doc
[Documentation]
include
[Header files]
lib
[Compiled libraries]
Source
[Source code from wich libraries are built]
```

4.2 Source

Note that a documentation of the source can be generated using the command

```
doxygen
```

in the moins directory. The documentation of the code is an ongoing effort, and any new function or class should be documented using the Doxygen format.

Follows a short summary of the main subdirectories of the Source directory:

4.2.1 Devices

This directory contains functions relative to specific hardware (e.g. Bumblebee, Staubli arm, etc.)

4.2.2 Features

This contains the classes defining the primitives and the metrics between them.

4.2.3 Image

Image functions, including signal processing and monogenic signal computation.

4.2.4 Matching

Old matching functions, most of the content is now deprecated

4.2.5 Mathematics

Contains a collection of mathematical tools and classes, like the Vector and

Matrix classes. Those classes are convenient as they allow for an elegant coding of mathematical relations, but they are poorly optimized.

4.2.6 MGraphics

Old graphic functions, all this ought to be deprecated, and progressively replaced with the use of the Cairo library. A demonstration of what this library could offer is available in

`public_demo/DisplayPrimitives/display_primitives`

4.2.7 Relations

Contains all relations that can be drawn between primitives, e.g. coplanarity, parallelism, etc. Some relations are still in Representation, and ought to be moved.

4.2.8 Representation

This is the core of the moins program, it is subdivided into two subdirectories:

scene: Where stereopsis and 3D reconstruction are implemented. The grouping of collinear segment and correction by interpolation of primitives is still there also, and would need to be moved to the Relations directory.

Object contains all temporal operations, and the accumulation over time of object representation.

4.2.9 tclap

This is an external library that we use to handle command line in the demos:

`http://tclap.sourceforge.net/`

4.2.10 Utilities

This contains a couple of useful functions and data structures.

4.3 Demo

[SK: Norbert said that this section should be dropped for good]

5 Doxygen Documentation

To use the documentation generated by DoxyGen, a configuration file must first exist. This can be created by executing the following command:

```
doxygen -g my_config_file
```

Then the configuration file can be edited to reflect your needs.
The doxygen documentation is then generated by executing the following command:

```
doxygen my_config_file
```

The documentation will be generated in the output directory specified in the configuration file, or by default in an html/ directory relative to the point where the doxygen command was executed. The documentation can be viewed by pointing your browser to the html/index.html file.

6 Demo Applications

All the pulic demos are available in the directory "MoInS/Demo/public_demo". These will have extensive documentation, readme and help files.

6.1 Capturing Image Sequences Demo

This section will be added shortly. It features a detailed description on recording images with the bumblebee camera.

6.2 Accumulation Demo

This demo can be used to generate 3D primitives, links and accumulated representations from a video sequence. public_demo/Accumulation/accumulation (for up-to-date information check the README file, the default.xml file or the generated doxygen documentation).

6.3 Rigid Body Motion Estimation Demo

Program for estimation a rigid body motion between two frames as described in [4].

6.4 Wanderer Demo

This Demo is temporarily located in MoInS/Demo/Wanderer2/ until it reaches a mature state. The wanderer demo can be used to display 3D primitives and the relations between the 3D primitives as well as initial reflexive grasping actions. The relations that can be displayed and analyzed currently include coplanarity, cocolority and a combination of both. The program allows selection of a certain Primitive and display of a certain relation only for that Primitive for better analysis.

The following keys can be used to manipulate the display:

Key	Action
'a'	Move forwards
'z'	Move backwards
'Left-arrow'	Move left
'Right-arrow'	Move right
'Up-arrow'	Move up
'Down-arrow'	move down

Moreover, the display area can be grabbed using the left-mouse button and can be rotated with the motion of the mouse. For up-to-date information check the README file, the config_example.xml or the generated doxygen documentation.

7 The ten Commandments

1) CoViS is a Group Effort and a Group Achievement. Everyone gains from Everyone. Therefore, think as a member of a Group that together is strong.

2) Do use English only in the Code and the Documentation.

3) Before you commit check Compilation and Running of Programs.

4) Provide Demo Programs for main Achievements.

5) Use speaking Names for Functions and Variables. Do a short comment for each Function and Variable.

6) Local Variables have to end by 'L', Parameters by 'P', global Variables by 'G', Class Variables by 'C'.

7) Use '.at( )' instead of '[ ]' for std vectors.

8) Use capital Letters for Constants and Enumerators

9) Call a Flag a Flag.

10) Do not use '.at[number]' but '.at(numerator)'.

8 Publications based on or related to CoViS

8.1 Publications

All publications, PhD and master theses can be founds at [6]

References

[1] Pacoplus: Perception, action and cognition through learning of object-action complexes. *Integrated Project*, 2006-2010.

[2] ECOVISION. Artificial visual systems based on early-cognitive cortical processing (EU–Project). *http://www.pspc.dibe.unige.it/ecovision/project.html*, 2003.

[3] http://www.pspc.dibe.unige.it/ drivsco/, editor. *DRIVSCO: Learning to Emulate Perception-Action Cycles in a Driving School Scenario (FP6-IST-FET, contract 016276-2)*. 2006-2009.

[4] N. Krüger and F. Wörgötter. Statistical and deterministic regularities: Utilisation of motion and grouping in biological and artificial visual systems. *Advances in Imaging and Electron Physics*, 131:82–147, 2004.

[5] N. Krüger and F. Wörgötter. Multi-modal primitives as functional models of hypercolumns and their use for contextual integration. *Proceedings of the 1st International Symposium on Brain, Vision and Artificial Intelligence 19-21 October, 2005, Naples, Italy, Lecture Notes in Computer Science, Springer, LNCS 3704*, pages 157–156, 2005.

[6] Cognituve Vision Lab. *www.mip.sdu.dk/covig*.

[7] F. Wörgötter, N. Krüger, N. Pugeault, D. Calow, M. Lappe, K. Pauwels, M. Van Hulle, S. Tan, and A. Johnston. Early cognitive vision: Using gestalt-laws for task-dependent, active image-processing. *Natural Computing*, 3(3):293–321, 2004.